Knowledge Distillation Scheme for Named Entity Recognition Model Based on BERT

Conference Paper · December 2023

DOI: 10.1109/MLBDBI60823.2023.10482264

2 authors, including:

Alexander A Hernandez

137 PUBLICATIONS **920** CITATIONS

SEE PROFILE

Knowledge Distillation Scheme for Named Entity Recognition Model Based on BERT

Ye Chengqiong[1,2]
1 College of Computing and Information Technologies, National University
Manila, Philippines
2 School of Big Data and Artificial Intelligence, Anhui Xinhua University
Hefei, China
yechengqiong@axhu.edu.cn

Alexander A. Hernandez*
College of Computing and Information Technologies, National University
Manila, Philippines
aahernandez@national-u.edu.ph

***Abstract*—In order to improve the applicability of named entity recognition model in industrial application scenarios, this paper studies the BERT named entity recognition task. From the perspective of lightweight model, an adaptive weight distillation method is designed to compress the BERT named entity recognition model. By adjusting the weight hyperparameter of distillation loss to an adaptive function, this method enables the model to automatically adjust the weight value according to the training process, avoiding the influence of manual parameter adjustment. Furthermore, by decreasing the middle hidden layer's dimension, this method compresses the teacher model from the width and incorporates the distillation of the middle Transformer layer of the model into the distillation process, allowing it to fully utilize the rich semantic information in the deep structure of the model. Ultimately, a light-weight, well-functioning model with a straightforward structure and few parameters is trained. By employing the suggested technique, the model's inference speed can be increased by seven times while reducing the number of model parameters to 15% of the original model with little loss of accuracy.**

Keywords- Knowledge distillation; Named entity recognition; BERT; Adaptive weight; Transformer

I. INTRODUCTION

Large pre-trained language models have shown strong performance on named entity recognition tasks in recent years. Nevertheless, because to the numerous parameters, lengthy training period, and high hardware resource requirements, these models are difficult to be applied to actual production. Therefore, it is crucial to reduce the storage and computation cost of such large pre-trained language models. In order to optimize deep neural network models and reduce their computational complexity and storage requirements, recent work has mainly focused on two directions: designing more efficient deep model architectures [1-2] and developing model compression and acceleration techniques [3]. The latter is separated even further into the subsequent groups: (1) Sharing and trimming of model parameters: By eliminating unnecessary parameters that don't significantly affect the model's performance, a process known as parameter pruning is used, which lowers the network's computational complexity and speeds up the model's inference. These techniques can be further broken down into parameter sharing, model quantization, model binarization, and other techniques. [4-8]. (2) Low-rank factorization: This type of method reduces the parameters of deep neural networks through matrix and tensor factorization [9-10]. (3) Knowledge distillation: This kind of process transfers information from a big deep neural network to a smaller one. By designating the original large deep neural network as the teacher model and the mini deep neural network as the student model, one can achieve a general framework for teacher-student distillation.

In image recognition, speech recognition, and recommendation systems, knowledge distillation is a frequently used technique for model compression that can reduce a complex model to a lightweight model with similar performance. Deep learning models, embodied by the pre-trained language model BERT [11], have demonstrated exceptional performance on several natural language processing tasks in the field of natural language processing. Nevertheless, these deep learning models frequently need a significant amount of time and resources. In recent years, more and more natural language processing tasks use knowledge distillation methods to obtain lightweight models, which effectively reduces the application cost of models. Taking machine translation tasks as an example, Rush and Kim [12] expanded machine translation situations from word-level knowledge distillation to sequence-level knowledge distillation. This method fits the sequential distribution of the teacher model with the sequential distribution of the student model. Tan et al. [13] proposed a multi-teacher distillation framework. In this framework, each independent teacher model is a monolingual model, and the student model is a multilingual model, which effectively handles the diversity of multilingual languages. In order to supervise the training of the student model and enhance the translation quality of the model, Freitag et al. [14] suggested an approach that used the combined output of numerous machine translation models as a teacher model. In order to improve the performance of machine translation and machine reading tasks, Wei et al. [15] proposed an online knowledge distillation method. This method selects the currently best evaluated model from the training process as the teacher model, and continuously updates it with better models obtained in subsequent evaluations, which can effectively deal with the instability in the training phase.

Deep, sophisticated, and having many parameters, BERT is a pre-trained language model that has garnered a lot of interest in the field of natural language processing, which is not easy to deploy. In view of this, researchers have conducted a large

number of lightweight studies on BERT using knowledge distillation [16-18]. A Patient Knowledge Distillation with BERT model compression (BERT-PKD) method was presented by Sun et al. [19] for machine reading comprehension, sentiment categorization, natural language inference, and paraphrase similarity matching. Jiao et al. [20] suggested TinyBERT, or two-stage Transformer [21] knowledge distillation, which combines task-specific and generic domain knowledge distillation to increase the speed of language inference. Tang et al. [22] suggested task-specific knowledge distillation from a BERT instructor model to a BiLSTM for sentence classification and matching.

Knowledge distillation started late in the field of named entity recognition, and compared with other fields, the above knowledge distillation methods solve some problems that need to be faced. However, in the process of knowledge distillation, the loss function is an important indicator to measure the feature proximity between teachers and students, which directly affects the effect of distillation. Therefore, how to design a more appropriate distillation loss objective function is a key problem to be solved. Considering the real-time requirements of model deployment and reasoning in actual algorithm application scenarios, this paper designed a knowledge distillation method with adaptive weights, and added the intermediate Transformer layer of BERT named entity recognition model to the distillation, so that the knowledge distillation process was closer to real learning behavior. The model lightweight is achieved with low accuracy loss.

II. ADAPTIVE KNOWLEDGE DISTILLATION SCHEME

Although the deep learning approach can frequently enhance the model's performance, it comes with a high network application cost and a poor inference time because of the vast number of deep learning network parameters. Bucilua et al. [23] developed a deep neural network model compression technique called knowledge distillation to address this issue. It transfers knowledge from a heavyweight model or integrated multiple models to a lightweight model without appreciably lowering the model's inference accuracy. The most common way to conduct knowledge distillation is through a teacher-student model framework, in which the student model is the smaller, simpler model and the teacher model is the larger, complex model.

Knowledge distillation is a procedure that is similar to educating students by a teacher with extensive experience: using the small and light student model, one fits the vast and complicated teacher model under the guidance of the teacher model. To be more precise, knowledge distillation normally involves two processes, as seen in Fig. 1: first, a large-scale teacher model is trained on the available data; next, a student model is trained using the dataset and the teacher model; and lastly, a lightweight student model is obtained. Utilizing the information of the student model to fit the knowledge of the instructor model while minimizing loss is the fundamental idea behind the knowledge distillation process.

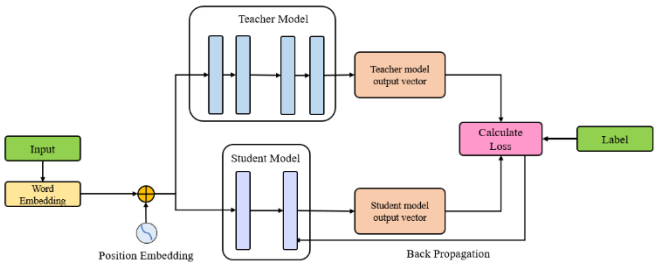

Figure 1. Framework of teacher-student model.

A. *Principle of knowledge distillation*

In knowledge distillation, the knowledge transfer process from the teacher model to the student model is as follows.The probability *pi* of the input belonging to class *i* can be approximated using the softmax function, as seen in (1), given the output vector *z* of the final fully connected layer of the neural network, also known as logits, where *zi* is the logit of class *i*.

$$p_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \tag{1}$$

Fig. 2 illustrates the difference between soft and hard labels. As an example of the handwritten digit recognition task in Fig. 2, the current input sample is the digit 2, that is, the value corresponding to the hard label 2 is 1, and the rest is 0. The soft label obtained by the model is a real-valued probability distribution, unlike the hard label, where the negative examples are all 0. The input sample on the left is closer to the number 3 than the sample on the right, so the soft label probability value corresponding to the number 3 is higher than the other ones, which indicates that the soft label actually contains a lot of useful information. Therefore, it is important for knowledge distillation to incorporate soft labels, and the model trained in this way can generalize better than the model trained with the same architecture but only using hard labels.

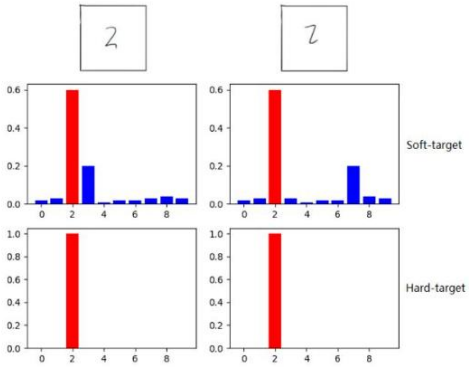

Figure 2. Examples of soft and hard labels.

In general, a softmax function is used to normalize the output scores after the model output to obtain a probability distribution in the mathematical sense. Such a real-valued probability distribution is often used to make soft labels. But in fact, if the obtained real-valued probability distribution entropy is small, this will make the probability distribution variance very large, that is to say, the value of the positive label is infinitely close to 1, and the others are infinitely close to 0, which will lead to the

model failing to converge. Therefore, it is common practice to introduce a temperature factor T that controls the importance of each soft label. The quality of knowledge transfer in the knowledge distillation process is closely related to the temperature factor T, as shown in (2).

$$p_i = \frac{\exp\left(\frac{z_i}{T}\right)}{\sum_j \exp\left(\frac{z_j}{T}\right)} \tag{2}$$

As the temperature factor T increases, the probability distribution of the predicted labels becomes smoother, and as T $\rightarrow$ 0, the soft labels degenerate into hard labels. The genuine hard label of the training sample and the soft label of the teacher model, which are used to compute the loss of the student model and the loss of the distillation, respectively, are crucial to enhancing the performance of the student model during the knowledge distillation process. The matching degree of logits between the teacher and student models is the distillation loss, sometimes referred to as the soft label loss, as shown in (3).

$$L_D\left(p(z_t,T),p(z_s,T)\right) = \sum_i -p_i(z_{ti},T)\, log\left(p_i(z_{si},T)\right) \tag{3}$$

Here, z_t and z_s are the logits of the teacher and student models, respectively. When the gradient is calculated, the student model's logits and the teacher model's logits have matching gradients., as shown in (4).

$$\frac{\partial L_D\left(p(z_t,T),p(z_s,T)\right)}{\partial z_{si}} = \frac{p_i(z_{si},T)-p_i(z_{ti},T)}{T} \tag{4}$$

When particularly high temperature T, $\frac{\partial L_D\left(p(z_t,T),p(z_s,T)\right)}{\partial z_{st}}$ can get (5) using Taylor series expansion.

$$\frac{\partial L_D\left(p(z_t,T),p(z_s,T)\right)}{\partial z_{si}} = \frac{1}{T}\left(\frac{1+\frac{z_{si}}{T}}{N+\sum_j \frac{z_{sj}}{T}} - \frac{1+\frac{z_{ti}}{T}}{N+\sum_j \frac{z_{tj}}{T}}\right) \tag{5}$$

Assuming that the training samples are zero-mean, that is, $\sum_j z_{sj} = \sum_j z_{tj} = 0$, then (5) can be simplified to (6).

$$\frac{\partial L_D\left(p(z_t,T),p(z_s,T)\right)}{\partial z_{si}} = \frac{1}{NT^2}\left(z_{si} - z_{ti}\right) \tag{6}$$

According to (6), assuming a high temperature factor T and zero mean logits, the distillation loss between the teacher and student models is equal to their respective logits differences. Therefore, highly useful information from the teacher model can be gathered and used to aid in the training of the student model by matching logits at high temperatures throughout the distillation process. Conversely, as (7) illustrates, the loss of the student model is the difference in value between the true and soft labels, or the hard label loss.

$$L_s\left(y,p(z_s,T)\right) = L_{CE}\left(y,p(z_s,T)\right) = \sum_i -y_i \log\left(p_i(z_{si},T)\right) \tag{7}$$

Where *y* is the one-hot vector of true labels. The same logits are used for both student model loss and distillation loss, but the temperature factor T is different. The temperature factor is 1 when calculating the student model loss, while the temperature factor is *t* when calculating the distillation loss. In summary, the loss of general knowledge distillation consists of two parts: the loss of student model and the distillation, as shown in (8):

$$L(x,W) = \alpha \times L_D\left(p(z_t,T),p(z_s,T)\right) + (1-\alpha) \times L_s(y,p(z_s,T)) \tag{8}$$

Where *x* is a training sample in the transfer set and *W* is a parameter of the student model.

B. Framework of adaptive knowledge distillation

Fig. 3 depicts the suggested adaptive knowledge distillation structure, which is split into two phases: First, a particular entity dataset is utilized to train the BERT model, and the fully trained model is employed as the distillation teacher model. The model then passes via knowledge distillation technology for lightweight compression, and relevant student model parameters are set.

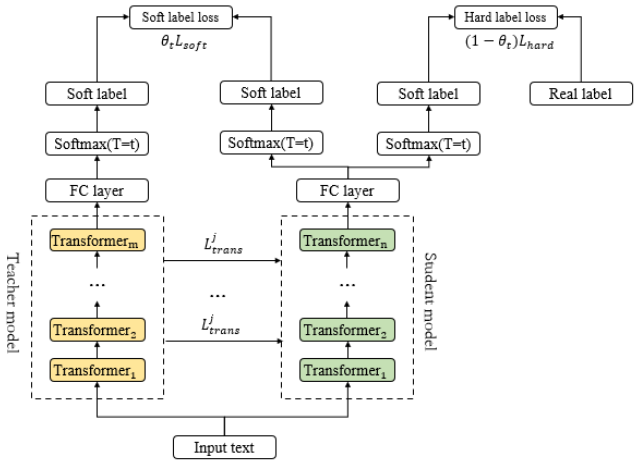

Figure 3. Framework for adaptive knowledge distillation.

In the specific implementation process of knowledge distillation, this paper improves the ordinary knowledge distillation method, and designs a knowledge distillation method based on adaptive weight, which is mainly optimized from two aspects of adaptive adjustment of hyper-parameters and intermediate distillation to ensure the performance and effect of the named entity recognition model after compression.

1) Adaptive weight method

The distillation target loss generally consists of two parts: hard label loss and soft label loss. In general, a weight parameter is set to balance the two losses, which is denoted as α. The ultimate loss of the model training is equal to the weighted sum of the two losses. In (9), the distillation loss function is provided.

$$L_{total} = \alpha L_{soft} + (1-\alpha) L_{hard} \tag{9}$$

In distillation, the loss function often takes the form of mean square error loss or cross-entropy loss. Among them, a popular loss function for determining the degree of correlation between two probability distributions is the Cross Entropy Loss., and the specific form is as follows (10):

$$CE = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log(p_i) + (1-y_i)\log(1-p_i)) \quad (10)$$

The Mean Squared Loss is computed as:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - p_i)^2 \quad (11)$$

In (9), L_{soft} represents the loss of soft label training, and the loss function chooses the cross-entropy loss. q_t is used to represent the soft labels obtained by the teacher model, and q_s is used to represent the predicted output of the student model, which is specifically expressed as (12):

$$L_{soft} = CE\big(softmaxT(q_t), softmax(q_s)\big) \quad (12)$$

L_{hard} represents the loss of the hard label part, as shown in (13), and y is the label of the real data.

$$L_{hard} = CE\big(y, softmax(q_s)\big) \quad (13)$$

From the form of the loss function, we can see that in the knowledge distillation process, we need to manually set the weight hyperparameter α of soft label loss and hard label loss. In fact, it is difficult to find the most appropriate parameter values by manually tuning the parameters, such as using grid search to determine the value of α, which is time-consuming and laborious, and it is also difficult to ensure the effect of distillation training. From the most essential point of view of knowledge distillation, our aim is to make the distillation process imitate human learning behavior as much as possible, not only learning from the output of the existing model, but also paying more attention to the process of learning knowledge, helping the model evolve from pure imitation to its own generalization.

Therefore, for the named entity recognition framework based on BERT model, this paper proposes a knowledge distillation method using adaptive weight learning. The purpose is to determine the value of the loss weight parameter α at different training times by the model itself in the process of knowledge distillation. Specifically, we replace the fixed parameter α with a dynamic function as shown in (14):

$$\theta_t = \max\left(1 - \frac{e^{\frac{t}{step}+b}}{e}, 0\right) \quad (14)$$

Where θ_t is the adaptive function that automatically adjusts the weights with different stages of model training, t is the step size of the current training of the model, and step is the total step size of the model training. b is the bias parameter, which can provide more space for the model training step. The value of the adaptive function is distributed in the interval 0-1. According to the characteristics of the adaptive function θ_t, in the initial training stage of the model, the step size is small, and the value of the distillation weight parameter θ_t starts from 1. Then with the increase of the step size of the training process, the value of the weight parameter is decreasing from 1 until it is close to 0 at the end of the training stage.

2) Transformer layer distillation

Using the adaptive function is important because it forces the student model to focus more on learning soft labels early in the training process, which improves the student model's capacity for generalization. The student model is trained to focus more on learning actual hard labels in the second half of the training, which increases prediction accuracy. After introducing the adaptive weight function, the loss function of knowledge distillation is adjusted to the following form:

$$L_{total} = \theta_t * L_{soft} + (1-\theta_t)L_{hard} + \theta_t \sum_{j=0}^{n} L_{trans}^{j} \quad (15)$$

In (15), n is the number of middle layers in the student model, and L_{trans} is the loss of the student model to the middle layer learning of the teacher model throughout the distillation process. The proposed adaptive knowledge distillation approach gains another significant improvement with the addition of the middle layer loss of the model.

The BERT model, which forms the bulk of the model in this study, is built on the Transformer structure. Its internal self-attention mechanism is crucial to the model's capacity to extract text features well. According to Clark et al. [24], the attention weights that BERT learns can represent extensive linguistic information knowledge, which is necessary for interpreting natural language and contains grammatical and anaphoric information. Consequently, the model information in the middle layer of the teacher model is ignored, and the student model does not truly accomplish the goal of learning, if the student model is trained solely using the soft labels and actual labels produced from the last layer of the BERT teacher model. In order to determine the distillation loss, this method additionally incorporates the middle layer Transformer component of the model into the distillation process. As illustrated in Fig. 4, the distillation of the intermediate layer consists of two components: the distillation of the feedforward neural network's output and the attention layer matrix. The following is an expression for the intermediate layer loss L_{trans}.

$$L_{trans} = L_{attn} + L_{hidn} \quad (16)$$

The formal definition of matrix distillation loss for intermediate attention layers is given in (17):

$$L_{attn} = \frac{1}{h}\sum_{i=1}^{h} MSE(A_i^S, A_i^T) \quad (17)$$

In (17), A_i^T is the attention weight matrix of the ith attention subspace in the teacher model; A_i^S is the attention weight matrix of the ith attention subspace in the student model; and h is the number of heads of the attention mechanism in the middle layer of the model. In particular, the unnormalized attention matrix A and the mean squared error (MSE) loss function are used as the loss form rather than the softmax normalized one since they produce better and faster convergence.

The middle layer attention in the BERT model training process actually contains a lot of information related to semantic and grammar, which is crucial to improve the generalization ability of the distillation model. In order to achieve the effect of mimicking the behavior of the teacher model and enable the student model to effectively transfer the various levels of

language knowledge learned by the BERT teacher model, the attention layer loss is added. This allows the student model to learn the information contained in the teacher model's attention layer. This is the definition of the distillation loss for a feedforward neural network with intermediary layers.

$$L_{hidn} = MSE(H^S W_h, H^T) \tag{18}$$

In (18), The output of the middle layer of the student model is represented by H^S, the output of the middle layer of the teacher model is represented by H^T, and the dimension transformation matrix W_h is used to unify the dimensions of the output of the middle layer of the two models. More specifically, during model distillation, W_h is automatically changed. When the student model's middle layer is initialized, a dimension that differs from the teacher model's middle layer is chosen. For instance, the student model is compressed from the width level and is set to 1200 dimensions, whereas the teacher model feedforward network layer has 3072 dimensions. Then, in order to guarantee that the dimensions of the two are consistent when calculating the loss, it is required to project H^S to the space where H^T is located and utilize W_h to transform the middle layer of the student model's dimension.

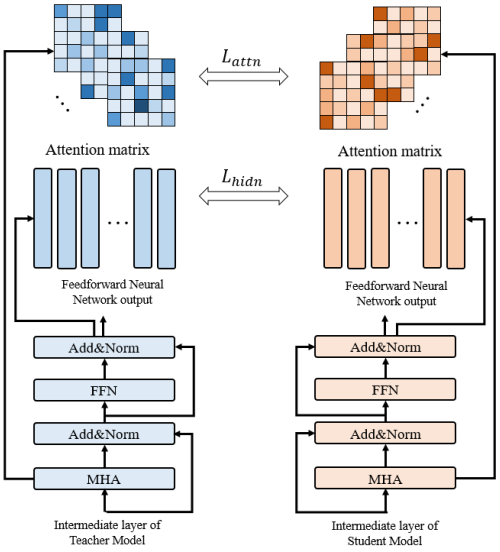

Figure 4. Diagram of distillation at Transformer layer.

When distilling the intermediate Transformer layer, this method uses the average linear mapping of the teacher-student intermediate layer. For example, The chosen instructor model is a 12-layer BERT model, as seen in Fig. 5. A 6-layer student model can be distilled by computing an intermediate layer loss every second layer, matching each student model layer to the instructor model's intermediate layer every second layer, and so on. The middle layer loss calculation at each layer includes the attention loss L_{attn} and the feedforward network loss L_{hidn}.

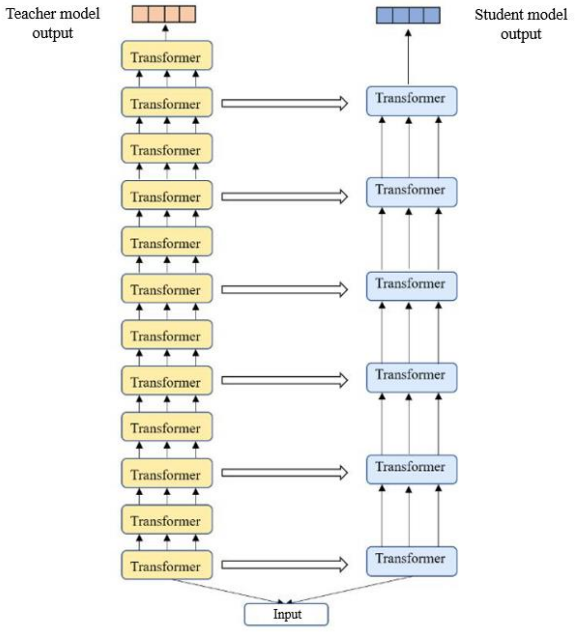

Figure 5. Average linear mapping of middle layer (taking layer 6 as an example).

In the concrete distillation process, this paper adopts a two-stage knowledge distillation scheme, which is used to initialize and train the student model, as shown in Fig. 6. Firstly, the BERT model that has not been fine-tuned by a specific data set is distilled to obtain an initialized lightweight student model. Then, the BERT teacher model is fine-tuned by using a specific named entity recognition data set. Finally, a lightweight student model was obtained, which was used in specific tasks.

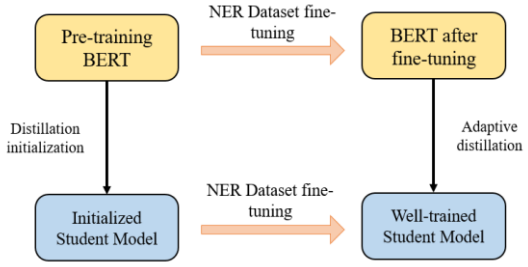

Figure 6. Student model initialization -fine-tuning distillation scheme.

III. EXPERIMENTS

A. Data Set

In this study, Note4 data set was used, which is a Chinese data set built by MIUI Company to identify four named entity types, including person (PER), place (LOC), animal (ANI) and plant (PLT). The data set adopted BIO annotation method.

In order to avoid the serious difference in model capability caused by data category tilt, category sampling is carried out during the generation of training data to balance the training of high-frequency and low-frequency entities, so as to avoid the situation that some high-frequency entities appear in the training data set, while some obscure entities only appear in a few times. The dataset is divided in an 8:1:1 ratio to obtain training,

validation, and testing sets, the detailed statistics of the Note4 dataset are shown in the following table:

TABLE I. NOTE 4 DATA EXAMPLE TABLE

Entity Type	Example	Number of Entities		
		Training set	*Validation set*	*Testing set*
PER	“姚明” (Yao Ming)	903726	180686	180535
LOC	“北京海淀” (Beijing Haiding)	797687	158528	158528
ANI	“狸花猫” (Lihuamao)	876358	175372	175416
PLT	“柑橘” (Ganju)	878688	175416	175415

B. Experimental environment and parameters

The operating system used in the experiment was Linux, the processor was Inter Core i7, the GPU was GeForce RTX 3090, and the framework was Pytorch. The model is based on the open-source, Chinese-language BERT-Base pre-trained model from Google, which has 12 Transformer layers and Huggingface checkpoint as its initialization.

In terms of external word vector, we choose the fully pre-trained Chinese word vector that is open source by Tencent AI Lab. This word vector is obtained through the text training of massive news and web pages by directional skip-gram[25] algorithm. The dimension of the word vector is 200, the size is 16G, and it covers more than 8 million Chinese words. The Chinese dictionary used is the dictionary corresponding to the word vector.

In terms of model setup, the word depth interaction module designed in this paper is applied between the Layer 1 and Layer 2 Transformer encoders of the BERT model, and BERT parameters and external word vectors are fine-tuned simultaneously during training.

In terms of experimental parameters, the maximum length of the Chinese text sequence is set to 128, and the batch size of the data set training is set to 64. For detailed parameter Settings, see the following table:

TABLE II. EXPERIMENTAL PARAMETER TABLE

Content	Parameter name	Numerical value
Input length	Maximum length of sequence	128
	Batch size	64
Pre-trained word vector	Dictionary size	8824330
	Word Vector Dimension	200
Bert-base	Number of network layers	12
	Hidden layer dimension	768
	Attention Headcount	12
	Parameter size	110M
Training	Epoch	20
	Learning rate	1e-5
	Dropout	0.1
	Optimizer	AdamW
	Random Seed	2021

C. Evaluation

The task's assessment index primarily concentrates on the model's F1 score on the data set, accuracy rate, and recall rate. When compared to the simple mean value calculation approach, the F1 score provides a thorough assessment of a model's performance. It is the harmonic balance between accuracy and recall rate. The calculation formulas of the three types of indicators are as follows:

$$Precision = \frac{TP}{TP+FP} \times 100\% \tag{19}$$

$$Recall = \frac{TP}{TP+FN} \times 100\% \tag{20}$$

$$F_1 = \frac{2 \times Precision \times Recall}{Precision+Recall} \times 100\% \tag{21}$$

Specifically, the counting rule of the sample result is in characters. When the character predicts the correct entity position and category, it is regarded as TP, and if it does not predict correctly, it is regarded as FP. It is not an entity class and is predicted to be 0 as TN, and it is not an entity class but is incorrectly identified as an entity as FN. In the multi-classification task, F1 is divided into Micro-F1 and Macro-F1. Macro-F1 calculates accuracy and recall rates for each entity type, and then averages them for all types. In contrast, Micro-F1 considers all entity types simultaneously to calculate the overall accuracy rate, recall rate, and F1 score as follows:

$$Precision_{Micro} = \frac{\sum_{i-1}^{n} TP_i}{\sum_{i-1}^{n} TP_i + \sum_{i-1}^{n} FP_i} \tag{22}$$

$$Recall_{Micro} = \frac{\sum_{i-1}^{n} TP_i}{\sum_{i-1}^{n} TP_i + \sum_{i-1}^{n} FN_i} \tag{23}$$

$$Micro - F_1 = 2 \times \frac{Precision_{Micro} \times Recall_{Micro}}{Precision_{Micro} + Recall_{Micro}} \tag{24}$$

In cases where the entity categories are not balanced, Micro-F1 can more accurately capture the entire model's impact. Therefore, in the named entity recognition experiments in this paper, the calculation method of Micro-F1 score is used as the evaluation index.

D. Model distillation experiment

In this paper, In order to confirm the efficacy of the knowledge distillation framework presented in this paper, the effectiveness of the model following distillation, and the impact of various student model layers, three sets of experiments are set up.

1) Validity verification of distillation algorithm

In this group of experiments, the effectiveness of knowledge distillation on named entity recognition task is compared between the lightweight model obtained by knowledge distillation and the lightweight model obtained by direct pruning. Secondly, the adaptive distillation method in this paper is compared with DistilBERT[26], BERT-PKD and TinyBERT and other knowledge distillation methods to verify the effect of adaptive knowledge distillation. In this group of experiments, the 12-layer BERT teacher model was compressed to 6 layers.

The initial student model was obtained by first distilling the unfine-tuned teacher model. The hidden layer dimension was set to 312 and the feedforward network dimension was set to 1200. Experimental results on the Note4 dataset are shown in the table below:

TABLE III. COMPARISON TABLE OF DISTILLATION ALGORITHM EFFECTIVENESS

Model	Accuracy	Recall	F1
Teacher model	91.75	92.37	92.04
6-layer BERT (Undistilled)	89.45	89.67	89.55
Distill BERT	89.65	90.12	89.87
BERT-PKD	89.45	90.89	90.16
TinyBERT	90.31	90.28	90.29
Proposed method	90.50	91.45	90.97

It can be seen from the experimental results that compared with the model directly cut to 6 layers, the last four lightweight models based on knowledge distillation technology in the table perform better on the named entity recognition task. Simple pruning of the model will cause the model to lose strong learning ability, and the model performance will be significantly reduced. Knowledge distillation can maintain the original model performance to a large extent while lightweight model.

Both TinyBERT and our method adopt a two-stage distillation method. According to experimental data, their performance in named entity recognition tasks is significantly better than the first three lightweight models obtained from knowledge distillation techniques, demonstrating the superiority of the two-stage distillation method. On the basis of the two-stage distillation method, this article adopts a dynamic function to automatically adjust the loss function to participate in Transformer layer distillation, which effectively preserves the performance of the original teacher model to a certain extent. The distilled lightweight model achieved better performance on the Note4 dataset, achieving a higher F1 value of 90.97%, retaining 98.84% of the original F1 value.

2) Comparison of model compression efficiency

In this collection of studies, the teacher model is compressed using various knowledge distillation techniques into a 6-layer student model. The model is then used to reason on the test set, and its operating speed is gauged by calculating the average reasoning time of a single item. The specific experimental results are as follows:

TABLE IV. COMPARISON TABLE OF MODEL COMPRESSION EFFICIENCY

Model	Layers	Hidden layer dimension	Dimension of feedforward neural networks	Parameter size	Speed
Teacher model	12	768	3072	110M	1.0ms
6-layer BERT (Undistilled)	6	768	3072	66M	2.1ms
DistilBERT	6	768	3072	67M	2.1ms
BERT-PKD	6	768	3072	66M	2.1ms
TinyBERT	6	768	3072	67M	2.1ms
Proposed method	6	312	1200	16.5M	7.0ms

The experimental results show that the 6-layer model created by directly reducing the number of layers or by utilizing other distillation techniques may be efficiently lowered to 60% of the original instructor model's size, while increasing the reasoning time to 2.1 times. This study presents a strategy that can compress the instructor model from two angles: width (the hidden layer dimension is decreased to 312 dimensions), which is more lightweight, and depth (compression into 6 layers). The reasoning speed is enhanced to seven times that of the instructor model, and the number of model parameters is decreased from the original 110M to 16.5M, or 15% of the original. More efficiently, the model inference speed is increased.

3) Verifying the influence of the number of layers in the student model

In this group of experiments, student models initialized with different numbers of Transformer layers were set for experiments. The number of Transforme layers was 2, 4 and 6 respectively, and the adaptive distillation method was used for training. Finally, the results were compared on the Note4 data set.

TABLE V. COMPARISON TABLE OF STUDENT MODEL LAYERS

Layers	Accuracy	Recall	F1
Teacher model (12 layers)	91.75	92.37	92.04
Distillation to 6 layers	90.50	91.45	90.97
Distillation to 4 layers	88.83	89.91	89.52
Distillation to 2 layers	86.87	87.32	87.08

Obviously, the student model with more layers is closer to the teacher model. Among them, the F1 value of the 6-layer model obtained by distillation is only about 1 percentage point difference from that of the teacher model, while the difference of the 2-layer model is close to 5 percentage points. Therefore, in knowledge distillation, there is a certain limit to the compression of the model, which requires comprehensive consideration of the performance and efficiency of the model combined with actual business application scenarios, and the selection of a lightweight model with higher efficiency and accuracy loss within an acceptable range.

IV. CONCLUSIONS

In this paper, an adaptive knowledge distillation method is designed to compress the BERT named entity recognition model. By adjusting the weight hyperparameter of distillation loss into an adaptive function, the model can automatically adjust the weight value according to the training process. The Transformer layer, which sits in the middle of the teaching model, is also added during the distillation process. Utilize all of the rich semantic data contained in the instructor model's deep structure. Additionally, the model is compressed from width by decreasing the dimension of the middle-hidden layer, and a lightweight student model with a straightforward structure, few parameters, and good performance is trained.

The findings of the experiment demonstrate that the student model can effectively mimic the teacher model's performance, keeping 98.84% of the teacher model's F1 score. Simultaneously, the student model's parameter scale is smaller,

taking up only 15% of the instructor model's parameter scale, and the model's reasoning speed is boosted seven times. In every one of the chosen data sets, the suggested technique outperforms the standard distillation model.

ACKNOWLEDGMENT

We get the support from the fund of Anhui Province Teaching Demonstration Course (2020SJJXSFK1294) and Anhui Province quality engineering project (2019kfkc158).

REFERENCES

[1] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510-4520.

[2] Ma, N., Zhang, X., Zheng, H. T., & Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proceedings of the European conference on computer vision (ECCV) . pp. 116-131.

[3] Cheng, Y., Wang, D., Zhou, P., & Zhang, T. (2018). Model compression and acceleration for deep neural networks: The principles, progress, and challenges. IEEE Signal Processing Magazine, 35(1), 126-136.

[4] Wu, J., Leng, C., Wang, Y., Hu, Q., & Cheng, J. (2016). Quantized convolutional neural networks for mobile devices. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4820-4828.

[5] Courbariaux, M., Bengio, Y., & David, J. P. (2015). Binaryconnect: Training deep neural networks with binary weights during propagations. Advances in neural information processing systems, 28.

[6] Sindhwani, V., Sainath, T., & Kumar, S. (2015). Structured transforms for small-footprint deep learning. Advances in Neural Information Processing Systems, 28.

[7] Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

[8] Wang, Y., Xu, C., Xu, C., & Tao, D. (2018). Packing convolutional neural networks in the frequency domain. IEEE transactions on pattern analysis and machine intelligence, 41(10), 2495-2510.

[9] Yu, X., Liu, T., Wang, X., & Tao, D. (2017). On compressing deep models by low rank and sparse decomposition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7370-7379.

[10] Denton, E. L., Zaremba, W., Bruna, J., LeCun, Y., & Fergus, R. (2014). Exploiting linear structure within convolutional networks for efficient evaluation. Advances in neural information processing systems, 27.

[11] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[12] Kim, Y., & Rush, A. M. (2016). Sequence-level knowledge distillation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 1317-1327.

[13] Tan, X., Ren, Y., He, D., Qin, T., Zhao, Z., & Liu, T. Y. (2019). Multilingual neural machine translation with knowledge distillation. In: Proceedings of the 2019 International Conference on Learning Representations.

[14] Freitag, M., Al-Onaizan, Y., & Sankaran, B. (2017). Ensemble distillation for neural machine translation. arXiv preprint arXiv:1702.01802.

[15] Wei, H. R., Huang, S., Wang, R., Dai, X., & Chen, J. (2019, June). Online distilling from checkpoints for neural machine translation. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers).pp. 1932-1941.

[16] Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. Advances in Neural Information Processing Systems, 33, 5776-5788.

[17] Liu, W., Zhou, P., Zhao, Z., Wang, Z., Deng, H., & Ju, Q. (2020). Fastbert: a self-distilling bert with adaptive inference time. arXiv preprint arXiv:2004.02178.

[18] Fu, H., Zhou, S., Yang, Q., Tang, J., Liu, G., Liu, K., & Li, X. (2021, May). LRC-BERT: latent-representation contrastive knowledge distillation for natural language understanding. In Proceedings of the AAAI Conference on Artificial Intelligence.Vol. 35, No. 14, pp. 12830-12838.

[19] Sun, S., Cheng, Y., Gan, Z., & Liu, J. (2019). Patient knowledge distillation for BERT model compression. EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference.

[20] Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., & Liu, Q. (2020). TinyBERT: Distilling BERT for natural language understanding. Findings of the Association for Computational Linguistics Findings of ACL: EMNLP 2020. https://doi.org/10.18653/v1/2020.findings-emnlp.372.

[21] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[22] Tang, R., Lu, Y., Liu, L., Mou, L., Vechtomova, O., & Lin, J. (2019). Distilling task-specific knowledge from bert into simple neural networks. arXiv preprint arXiv:1903.12136

[23] Buciluǎ, C., Caruana, R., & Niculescu-Mizil, A. (2006, August). Model compression. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 535-541.

[24] Clark, K. , Luong, M. T. , Le, Q. V. , & Manning, C. D. . (2020). Electra: pre-training text encoders as discriminators rather than generators. arXiv preprint arXiv:2003.10555, 2020.

[25] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[26] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108, 2019.